Distros, Packaging, and Compiling

UNIX SysAdmin DeCal Fall 2020
Ethan Smith
(with thanks to Tony Situ)

Topics

- Intro to Distributions/Debian
- Packaging
- Compilation

Linux Distributions and Debian

Distributions?

- Basically the Linux kernel + other software = operating system
- Because there are many different configurations of the kernel and other software, Linux OSes have a term called “distributions”
- Are spun off a lot. “Derivatives”
  - Debian -> Ubuntu, RHEL -> Scientific Linux, Arch -> Manjaro

Debian

- Debuted in late 1993
- Why use it?
  - Stable - new release every 2 yrs.
  - User-friendly - works out-of-the-box
  - Respects your privacy (cough cough pre-16.04 vanilla Ubuntu cough)
- ~~Cute naming scheme~~
- Widely used

Installing software: package managers

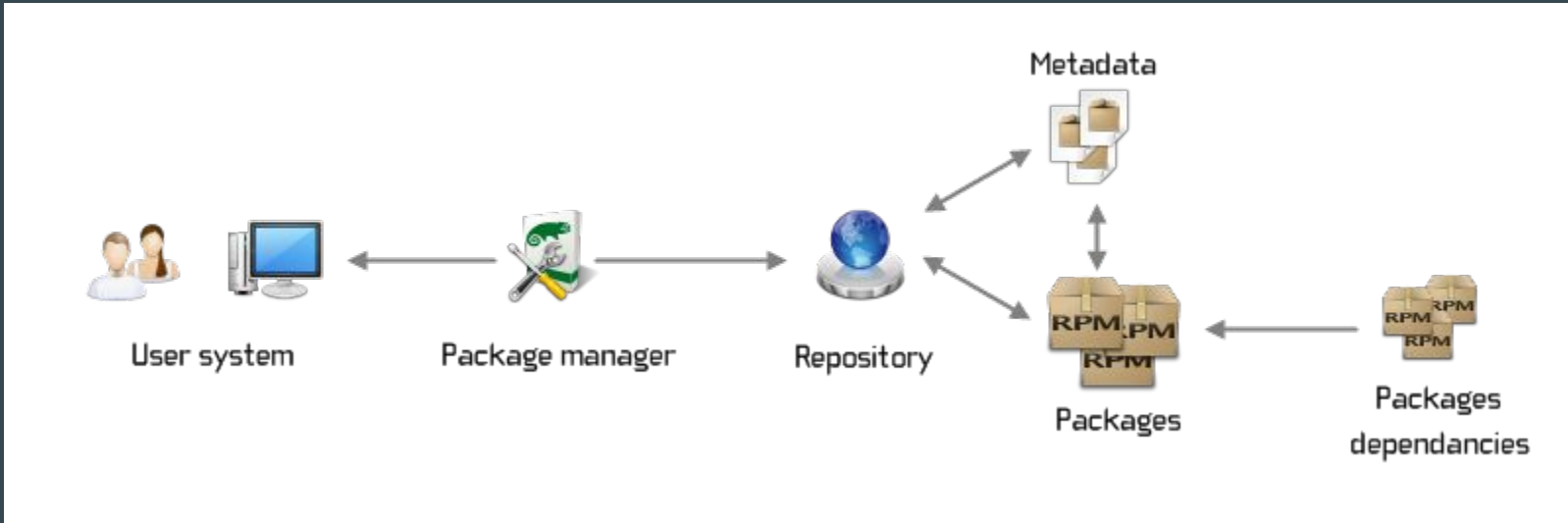
What is a package?

- An archive containing binaries and libraries of an application
- Also includes some other metadata for the system about the application
- Used to install new applications onto a system
- Debian uses the `.deb` format

Installing packages from file

- WARNING: installing from file <-> installing from .exe/.msi in Windows
  - Repositories == secure packages
  - Use common sense! Trusted sources
  - Ubuntu Personal Package Archives (PPA) have same risk

What's a package manager?



Why?

- Updates & security patches distributed fast
- Packages reviewed for malware
- No need to manually configure options
- Fast, easy way to install/update for user

Wait how do I use one then?

```
$ apt update
```

- Grabs a new list (catalogue) of what packages are available.

```
$ apt install <packagename>
```

- Installs the package.

```
$ apt remove <packagename>
```

- Uninstalls the package.

Wait how do I use one then?

```
$ apt dist-upgrade
```

Updates the packages and resolves package conflicts/removals.

```
$ apt search <packagename>
```

Searches for the package in the catalogue

```
$ apt install ./package.deb
```

```
Install local package (dangerous!)
```

A Demonstration of Package Installation

Install random package (cowsay).

Manually Compiling Software

- Usually packages downloaded from the web have a **Makefile** .
- Basically they make the application for you as long you have the right things installed (like compilers like GCC).
- First $ `./configure`
- Just run `make` and then `make install`.
- Can be pain in the ass to remove -- this is why package managers exist!

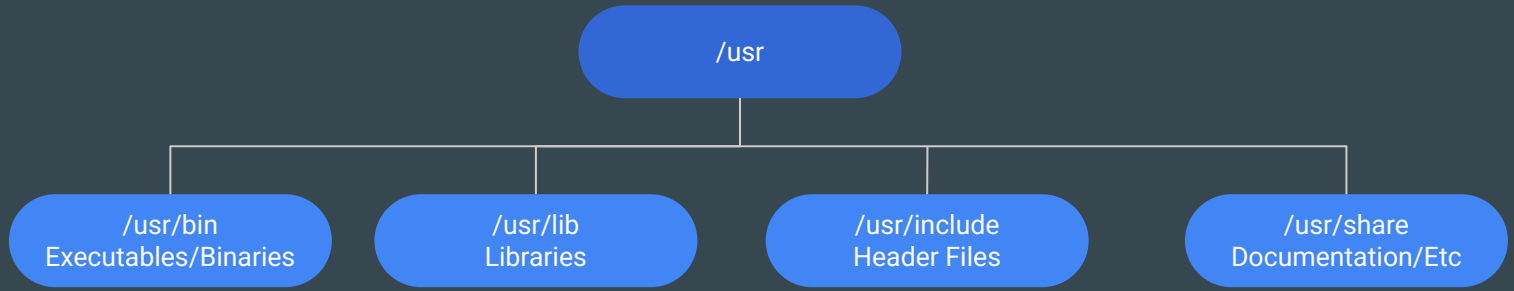
What is Compilation?

- Turns source code into a real executable.
- Turns ingredients into a dish.
- `$ gcc hellopenguin.c -o hellopenguin`

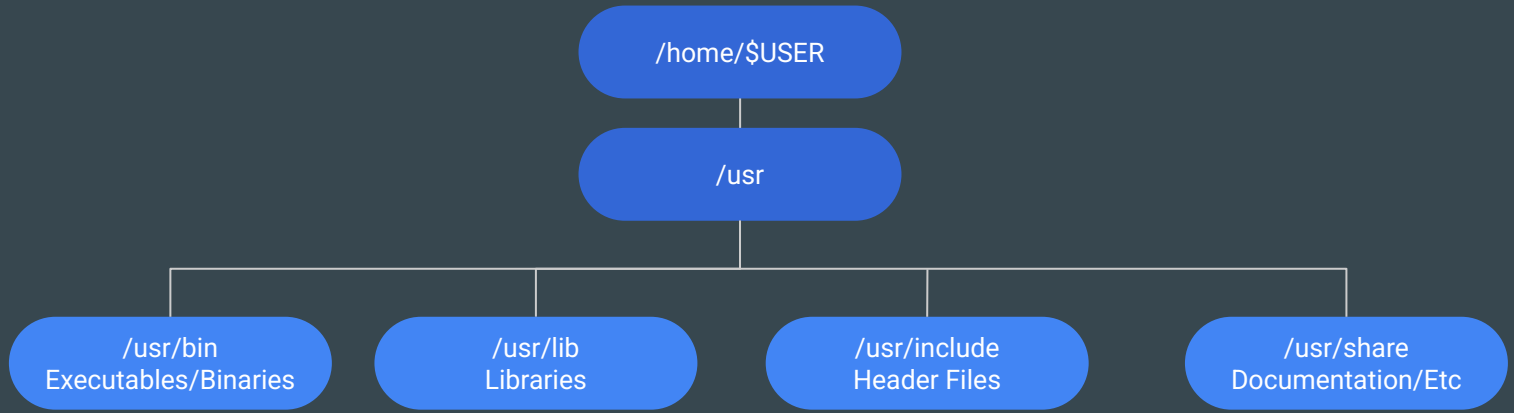
Ok now, how do I make a package?

- Huge pain in the ass
  - Download software/write it
  - Make sure everything configured
  - List dependencies, metadata (version, author, etc.)
  - Make sure linked libraries are in place
  - Use tool to make package

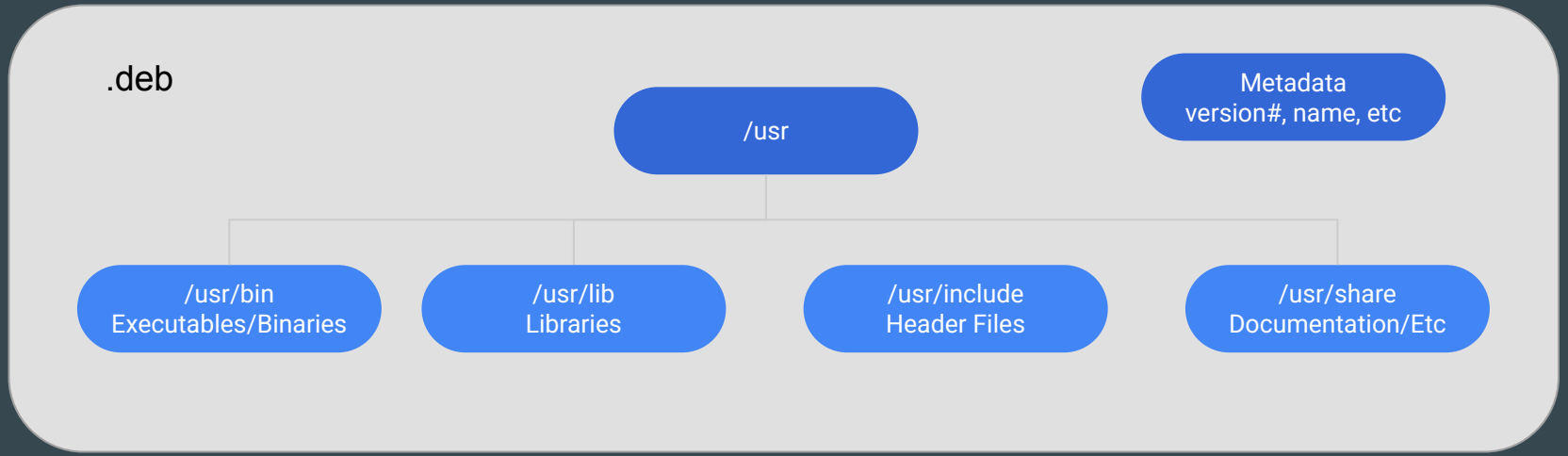
Packaging Basics - Installed layout



Packaging Basics - Package source layout



Packaging Basics - Package layout



Packaging

- In the lab we're going to use Effing Package Management (FPM), which is a Ruby Gem and makes packaging (slightly) less painful
- `$ sudo apt install ruby-dev`
- `$ sudo gem install fpm`
- `$ fpm -s dir -t deb -n [name here] -v [version #] -C [the directory with the /usr folder]`

More Resources?

Google

FPM documentation

- https://github.com/jordansissel/fpm/wiki

Debian documentation

- https://www.debian.org/doc/

Assignment

- decal.xcf.sh/labs/b4
  - Install gcc, ruby-dev, fpm
  - Write your own “hellopenguin” program, compile it, then package it, then install it.